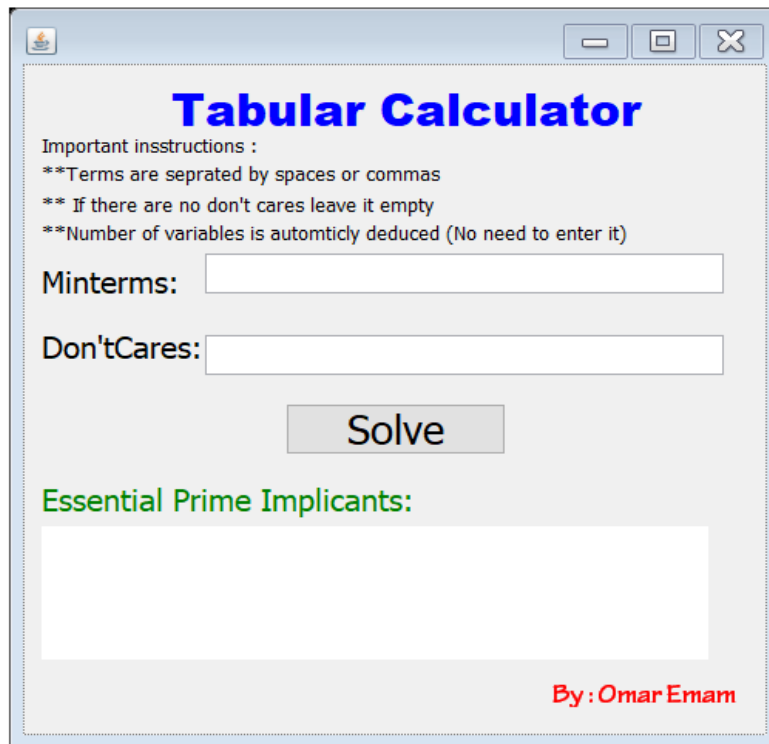


# TABULAR CALCULATOR

## REPORT



The screenshot shows a Windows-style application window titled "Tabular Calculator". Inside the window, the title "Tabular Calculator" is displayed in blue. Below it, there are "Important instructions" in black text: "\*\*Terms are separated by spaces or commas", "\*\* If there are no don't cares leave it empty", and "\*\*Number of variables is automatically deduced (No need to enter it)". There are two input fields: "Minterms:" and "Don'tCares:". Below these fields is a "Solve" button. Under the button, the text "Essential Prime Implicants:" is shown in green, followed by a large empty text area for the results. At the bottom right of the window, the text "By: Omar Emam" is displayed in red.

## CONTENTS:

1. PROBLEM STATEMENT
2. PROGRAM DESCRIPTION
3. PROGRAM FLOW
4. USED DATA STRUCTURES
5. SAMPLE RUNS

BY: OMAR MOHAMED EMAM (47)

## 1 . PROBLEM STATEMENT:

-Create a program that takes the min-terms and don't cares and then calculate the essential prime implicants using Quine-McCluckey algorithm (tabular method).

## 2. PROGRAM DESCRIPTION:

-the program scan min-terms and don't cares without the number of variables as the program deduce the number of variable from the binary representation for the biggest term and after solving the problem (as described in the flow) and then print the essential PI's and the minimum expression.

-the program is implemented in java and has a GUI implemented with java.fx package.

-the program has features to avoid all types of error like (typing errors or logic errors [same term in min-terms and don't care !!])

-you could review the source code on github at this link :

<https://github.com/omarmohamedemam/Tabular-Calculator>



### 3. PROGRAM FLOW:

- the program scans min-terms and Dc then, check their validation (that all inputs are only numbers (through is valid fun.) and also there are no repeated terms (through check fun.) )and then convert them to binary representation.
- Combine min-terms and Dc in one array and sort them according to the number of one in the binary representation.
- Divide the Terms in groups according to the number of one in the binary representation.
- Start the steps of solving:
  - there are three main loops
    1. The outer one loops on the groups
    2. The middle loops on the first group elements
    3. The inner loops on the second group elements
  - every time it check if the combination between two elements is valid (through comparing each bit in the two elements and return true in case of the difference is only one bit (through is valid combination fun.)
  - Then, check if this combination is already taken or not if not, add it to the taken combinations.
  - After finishing combining for the first time add the untaken terms to the next step.
  - Repeat these steps till all elements are checked (moved) and there are no other possible combinations.

-The second Step of solving is to find the prime implicants through:

-identify prime:

This function construct the PI'S table in 2D array and check if the essential (repeated once) covers all the min-terms (are included in the row) or not

-if the previous function failed to get the essential PI's , the second function (Row dominance)

identifying the dominated rows and delete them from the table and hence apply the previous check

- if the previous function failed to get the essential PI's , the third function (Petrick) by constructing a table for each term and where is it repeated and convert them to expression and multiply the terms and finally get the minimum terms using (petrick's key) through using its length and bits

-Once you get the essential PI'S you should save them and convert them to an expression using (to symbolic function) as it converts (0, 1,-) to a letter according to its place in the bit and finally print the output.

#### 4. USED DATA STRUCTURES:

- {TERM} contains : string contains the number, the number of ones , the number representation in binary
- Also contains two constructors:
  1. To identify the term , convert it to binary , get the length and deduce the variables number , count the ones for the term
  2. For two terms it compares them, replace the common bit (0/1) by (-), recount the ones, re-initialize the term.

-Used in the rest of the program arrays, array lists and hash sets.

#### 5. SAMPLE RUNS:

-the program is able to solve the maximum possible number of variables (which is 28) due to the representation of variables in the alphabet.

(3 Variables):

The screenshot shows a window titled "Tabular Calculator" with a light gray background. At the top, there are window control buttons (minimize, maximize, close). Below the title, the text "Important insstructions :" is followed by three instructions: "\*\*Terms are seprated by spaces or commas", "\*\* If there are no don't cares leave it empty", and "\*\*Number of variables is automtically deduced (No need to enter it)". There are two input fields: "Minterms:" with the value "0,1,2,3,5,6" and "Don'tCares:" which is empty. A blue "Solve" button is centered below the input fields. Underneath the button, the text "Essential Prime Implicants:" is displayed in green. Below this, the output shows "0-- + -01 + -10" and "(A' + B'C + BC')". At the bottom right, the text "By : Omar Emam" is written in red.

**Tabular Calculator**

Important insstructions :  
\*\*Terms are seprated by spaces or commas  
\*\* If there are no don't cares leave it empty  
\*\*Number of variables is automtically deduced (No need to enter it)

Minterms: 0,1,2,3,5,6

Don'tCares:

Solve

Essential Prime Implicants:

0-- + -01 + -10  
(A' + B'C + BC')

By : Omar Emam

(4 Variables):

The screenshot shows a window titled "Tabular Calculator" with a light gray background. At the top, there are window control buttons (minimize, maximize, close). Below the title, the text "Important insstructions :" is followed by three instructions: "\*\*Terms are seprated by spaces or commas", "\*\* If there are no don't cares leave it empty", and "\*\*Number of variables is automtically deduced (No need to enter it)". There are two input fields: "Minterms:" with the value "0,1,5,7,13,15" and "Don'tCares:" which is empty. A blue "Solve" button is centered below the input fields. Underneath the button, the text "Essential Prime Implicants:" is displayed in green. Below this, the output shows "000- + -1-1" and "(A'B'C' + BD)". At the bottom right, the text "By : Omar Emam" is written in red.

**Tabular Calculator**

Important insstructions :  
\*\*Terms are seprated by spaces or commas  
\*\* If there are no don't cares leave it empty  
\*\*Number of variables is automtically deduced (No need to enter it)

Minterms: 0,1,5,7,13,15

Don'tCares:

Solve

Essential Prime Implicants:

000- + -1-1  
(A'B'C' + BD)

By : Omar Emam

(4Variables with don't care):

**Tabular Calculator**

Important instructions :  
\*\*Terms are separated by spaces or commas  
\*\* If there are no don't cares leave it empty  
\*\*Number of variables is automatically deduced (No need to enter it)

Minterms:

Don'tCares:

Essential Prime Implicants:

-0-- + 1---  
(B' + A)

By : Omar Emam

(5Variables):

**Tabular Calculator**

Important instructions :  
\*\*Terms are separated by spaces or commas  
\*\* If there are no don't cares leave it empty  
\*\*Number of variables is automatically deduced (No need to enter it)

Minterms:

Don'tCares:

Essential Prime Implicants:

-00-- + 00-10 + 1110-  
(B'C' + A'B'DE' + ABCD')

By : Omar Emam

(10Variables with don't care):

**Tabular Calculator**

Important insstructions :

- \*\*Terms are seprated by spaces or commas
- \*\* If there are no don't cares leave it empty
- \*\*Number of variables is automticly deduced (No need to enter it)

Minterms:

Don'tCares:

**Solve**

**Essential Prime Implicants:**

1011101110 + 1101111010  
(AB'CDEF'GHIJ' + ABC'DEFGH'IJ')

By : Omar Emam

**PROVIDED BY:**

**OMAR MOHAMED EMAM**